Actividades

1) Contestar las siguientes preguntas utilizando las guías y documentación proporcionada (Desarrollar las respuestas) :

• ¿Qué es GitHub?

Primero es necesario definir GIT. Es un herramienta de control de versiones creada en 2005 por Linus Torvalds. Este software especializado permite hacer un seguimiento de los cambios en un proyecto, quién los realizó, cuándo se realizaron, qué conflictos surgieron y poder volver para atrás. Git crea un repositorio de nuestros proyectos a nivel local.

Github es una comunidad donde podemos compartir nuestros repositorios de forma pública o privada. Una vez creada una cuenta en Github vamos a disponer de un perfil símil a una red social donde podremos cargar nuestros repositorios, seguir a otros usuarios, ver los repositorios de otras personas, clonarlos, guardarlos en favoritos, etc.

• ¿Cómo crear un repositorio en GitHub?

Dentro de nuestro perfil de GITHUB. Seleccionamos el + que se encuentra en la parte superior izquierda. Ahí seleccionamos NEW REPOSITORY. Es una buena práctica ponerle al repositorio el nombre de la carpeta local donde lo guardamos. Podemos colocar una descripción opcionalmente. Podemos elegir si será de acceso público o privado. Finalmente, presionamos en CREAR REPOSITORIO.

• ¿Cómo crear una rama en Git?

Cuando comenzamos a trabajar, lo hacemos en la rama principal llamada main o master (dependiendo de la versión). Si queremos crear una nueva debemos proceder así: dentro de la consola debemos utilizar el comando *git branch nombre_rama*.

• ¿Cómo cambiar a una rama en Git?

Supongamos que quiero acceder a la rama llamada nombre_rama. En la consola debemos escribir el comando *git checkout nombre_rama*.

• ¿Cómo fusionar ramas en Git?

En algunos casos, debemos fusionar dos ramas en una sola. Esto es hacer un merge (unir en español). Para realizar un merge el primer paso es pararme sobre la rama donde quiero que los cambios queden guardados.

>git checkout main (en este caso, voy a fusionar en la rama main).

A continuación, realizo la fusión:

>git merge rama2 (Así, la rama2 se fusionará con la main y los cambios quedarán asentados en la rama main).

• ¿Cómo crear un commit en Git?

Cuando ocurre algún cambio significativo en el proyecto (añadir, borrar o editar algún archivo), podemos marcar un punto en ese momento para volver o simplemente documentar el cambio. Ese punto se llama “COMMIT” (cometido en español). Suele estar identificado con algún mensaje puesto por nosotros para saber qué hicimos.

Además GIT tiene por detrás un código HASH que podemos consultar para llevar un registro unívoco de los cambios hechos. Es preferible tener varios COMMITS específicos en cada cambio y no COMMITS muy abarcativos para no perder cambios que no queremos.

• ¿Cómo enviar un commit a GitHub?

Primero debemos tener los cambios guardados en GIT a nivel local. Una vez hecho esto, en la consola escribimos:

>git remote add origin http://la_dirección_de_nuestro_repositorio (esto se hace una sola vez para diferenciar donde está nuestro repositorio remoto).

• ¿Qué es un repositorio remoto?

Un repositorio remoto es una versión de un repositorio de código que se encuentra almacenada en un servidor o en la nube, accesible desde múltiples ubicaciones. Permite a los desarrolladores colaborar y compartir su código, hacer "commits" y mantener versiones sincronizadas sin necesidad de estar en el mismo entorno físico. GitHub, GitLab y Bitbucket son ejemplos populares de plataformas que gestionan repositorios remotos.

• ¿Cómo agregar un repositorio remoto a Git?

Inicializamos el repositorio local: *>git init*

Luego, agregamos el repositorio remoto usando el comando git remote add, seguido del nombre del remoto (usualmente origin) y la URL del repositorio remoto:

>git remote add origin direccion-del-repositorio

A continuación, colocamos la siguiente instrucción:

>git push -u origin master (en este punto puede llegar a pedirnos las credenciales de GITHUB).

Actualizamos GITHUB y ya deberíamos ver las actualizaciones.

• ¿Cómo empujar cambios a un repositorio remoto?

Explicado en el punto anterior.

• ¿Cómo tirar de cambios de un repositorio remoto?

Primero asegurarnos de estar en la rama correcta:

>git branch

Tirar los cambios del repositorio remoto: usamos el comando git pull para obtener y fusionar los cambios del repositorio remoto a tu rama local:

>git pull origin main

origin es el nombre del repositorio remoto (por defecto), y *main* es la rama del repositorio remoto desde la cual estás tirando los cambios. Si estás trabajando con otra rama, reemplázala por su nombre correspondiente.

• ¿Qué es un fork de repositorio?

Un fork de un repositorio es una copia independiente de un repositorio que se crea en nuestra cuenta de una plataforma de control de versiones como GitHub. Al hacer un fork, podemos realizar cambios en el código sin afectar el repositorio original. Este proceso es útil para contribuir a proyectos de código abierto porque podemos hacer un fork de un repositorio, modificarlo y luego enviar esos cambios al repositorio original mediante un pull request (y ver si es aceptado o no). Además, podemos experimentar y desarrollar sin riesgos ya que tenemos la libertad de realizar cambios en nuestro fork sin afectar el trabajo de otros colaboradores.

• ¿Cómo crear un fork de un repositorio?

Dentro de GITHUB en la dirección del repositorio a copiar, hacemos click en FORK. Ponemos una descripción y usamos el botón CREATE FORK.

• ¿Cómo enviar una solicitud de extracción (pull request) a un repositorio?

Ir al repositorio en GitHub, encontraremos un botón para "Compare & pull request" en la página de la rama recién subida. Hacer clic en ese botón. En la página del pull request, describimos los cambios que realizamos y seleccionamos la rama a la que queremos hacer la solicitud de extracción (usualmente la rama principal del repositorio original). Enviamos la solicitud, hacemos clic en el botón Create pull request para enviarla.

• ¿Cómo aceptar una solicitud de extracción?

Si estamos listos para aceptar los cambios, hacemos clic en el botón Merge pull request. GitHub nos mostrará una confirmación de que queremos combinar las ramas. Si todo está bien, hacemos clic en Confirm merge para completar el proceso.

• ¿Qué es un etiqueta en Git?

En Git, una etiqueta (o tag) es una referencia que apunta a un punto específico en la historia del repositorio, **(**generalmente a un commit) para marcar un hito importante en el desarrollo del proyecto. Las etiquetas son comúnmente usadas para marcar versiones importantes, como una nueva versión de un software, un lanzamiento (release) o un estado específico del código que se quiere conservar para futuras referencias.

Existen dos tipos principales de etiquetas en Git:

- Etiquetas ligeras (lightweight tags): son como un puntero a un commit específico, no contienen más información que el commit al que apuntan, es una etiqueta simple sin metadatos adicionales.
- Etiquetas anotadas (annotated tags): son más completas, ya que contienen más información, como el nombre del creador, la fecha, un mensaje de anotación y una firma GPG si es necesario. Son más adecuadas para marcar versiones oficiales o puntos importantes en el proyecto.

• ¿Cómo crear una etiqueta en Git?

Etiqueta ligera:

>git tag nombre-de-la-etiqueta

Etiqueta anotada:

>git tag -a nombre-de-la-etiqueta -m "Mensaje descriptivo"

• ¿Cómo enviar una etiqueta a GitHub?

Para subir una etiqueta al repositorio remoto, usa:

>git push origin nombre-de-la-etiqueta

Si quieres subir todas las etiquetas de una vez:

>git push –tags

• ¿Qué es un historial de Git?

El historial de Git es el registro completo de todos los cambios realizados en un repositorio a lo largo del tiempo. Este historial incluye información sobre cada commit, como qué cambios se hicieron, cuándo se hicieron, quién los hizo y, en algunos casos, un mensaje descriptivo que explica el propósito del cambio.

Cada commit en Git tiene un identificador único (hash), lo que permite rastrear y comparar diferentes versiones del código. Además, el historial de Git también almacena las ramas y las fusiones (merges) que se han hecho a lo largo del tiempo.

El historial permite a los desarrolladores ver la evolución del proyecto, revertir cambios, comparar versiones o encontrar errores específicos introducidos en el tiempo. Puedes ver el historial usando el comando git log.

• ¿Cómo ver el historial de Git?

Podemos ver el historial de Git usando el comando git log.

• ¿Cómo buscar en el historial de Git?

-Si deseas buscar commits que contengan un mensaje específico, usamos:

>git log --grep="mensaje" (Esto buscará en los mensajes de los *commits* que coincidan con el término de búsqueda).

-Si quieres ver los *commits* que afectaron a un archivo específico, usamos:

>git log -- <ruta-del-archivo>

-Para buscar los *commits* hechos por un autor específico:

>git log --author="nombre del autor"

-Buscar un commit específico por hash:

>git show <commit-hash>

-Filtrar por fechas: Podemos buscar los commits realizados en un rango de fechas utilizando la opción --since y --until:

>git log --since="2025-03-01" --until="2025-03-30"

• ¿Cómo borrar el historial de Git?

Usamos el comando

>git reset

• ¿Qué es un repositorio privado en GitHub?

Un repositorio privado en GitHub es un repositorio cuyo acceso está restringido a usuarios específicos que tú autorices. A diferencia de un repositorio público, donde cualquier persona puede ver, clonar y contribuir al código, en un repositorio privado solo los colaboradores con permisos pueden acceder al contenido, realizar cambios o gestionar el repositorio. Esto lo hace ideal para proyectos que no deseas compartir públicamente, como trabajos privados, investigaciones o código confidencial. Los repositorios privados ofrecen mayor seguridad y control, ya que su visibilidad está limitada a los usuarios que tú elijas, asegurando que el contenido no sea accesible a personas no autorizadas. Para crear uno, simplemente seleccionas la opción "Private" al crear el repositorio en GitHub.

• ¿Cómo crear un repositorio privado en GitHub?

Al momento de crear un repositorio seleccionamos la visibilidad privada.

Bajo la sección "Visibility", selecciona la opción "Private" para que el repositorio sea privado.

• ¿Cómo invitar a alguien a un repositorio privado en GitHub?

Abrimos la configuración del repositorio:

- En la página del repositorio, haz clic en la pestaña "Settings" (Configuración), que se encuentra en la parte superior del repositorio.

2. Accedemos a la sección de colaboradores:
    - En el menú de la izquierda, seleccionamos "Manage access" (Gestionar acceso) bajo la sección "Access".
3. Invitamos colaboradores:
    - Hacemos clic en el botón "Invite a collaborator" (Invitar a un colaborador).
4. Introducimos el nombre de usuario de GitHub:
    - Escribimos el nombre de usuario de GitHub de la persona a la que deseas invitar. GitHub sugerirá coincidencias a medida que escribes.
5. Enviamos la invitación:
    - Después de seleccionar el usuario, hacemos clic en "Add" o "Invite" para enviar la invitación.

La persona recibirá una notificación por correo electrónico y podrá aceptar la invitación para obtener acceso al repositorio privado. Una vez aceptada, tendrá acceso según los permisos que le hayas asignado.

• ¿Qué es un repositorio público en GitHub?

Un repositorio público en GitHub es un repositorio cuyo contenido es accesible para cualquier persona. Cualquier usuario, incluso aquellos que no tengan cuenta en GitHub, puede ver, clonar, bifurcar (fork) y contribuir al proyecto. Los repositorios públicos son ideales para proyectos de código abierto, donde se desea compartir el trabajo con la comunidad para que otros puedan colaborar, revisar el código, hacer mejoras o aportar sugerencias.

Además, los repositorios públicos pueden ser indexados por motores de búsqueda, lo que significa que el contenido del repositorio puede aparecer en los resultados de búsqueda. Los desarrolladores usan repositorios públicos para compartir bibliotecas, herramientas, documentación o proyectos en los que buscan la colaboración o simplemente quieren que el código esté disponible para todos.

Cuando creas un repositorio en GitHub, la opción por defecto es público, aunque puedes cambiar esta visibilidad en cualquier momento si decides que el repositorio sea privado.

• ¿Cómo crear un repositorio público en GitHub?

Al crear un repositorio seleccionamos la visibilidad:

- Bajo la sección "Visibility", seleccionamos "Public" para que el repositorio sea accesible a cualquier persona.

• ¿Cómo compartir un repositorio público en GitHub?

Copiar la URL del repositorio y compartir el enlace. Una vez que hayas copiado la URL del repositorio, podemos compartirla directamente por correo electrónico, redes sociales o cualquier otra plataforma que desees.